PadhAI: Backpropagation - the full version

One Fourth Labs

Computing derivatives w.r.t Hidden Layers
Part 2

1. We have $\frac{\partial L(\theta)}{\partial h_{ij}} = (W_{i+1,\,\cdot\,,j})^T \nabla_{a_{i+1}} L(\theta)$
    a. This is with respect to one neuron
    b. We would like to speed up this computation by solving all the derivatives in one go
2. We can now write the gradient w.r.t h_i
    a.

$$\nabla_{h_i} L(\theta) = \begin{bmatrix} \frac{\partial L(\theta)}{\partial a_{h_{i1}}} \\ \cdot \\ \cdot \\ \cdot \\ \frac{\partial L(\theta)}{\partial h_{in}} \end{bmatrix}$$

$$\nabla_{h_i} L(\theta) = \begin{bmatrix} (W_{i+1,\,\cdot\,,1})^T \nabla_{a_{i+1}} L(\theta) \\ \\ \cdot \\ \cdot \\ \cdot \\ \\ (W_{i+1,\,\cdot\,,n})^T \nabla_{a_{i+1}} L(\theta) \end{bmatrix}$$

    b. Can be written more compactly as $(W_{i+1})^T \nabla_{a_{i+1}} L(\theta)$
3. Thus, the formula for gradient of loss function for the last hidden layer before the output layer is given by $\nabla_{h_i} L(\theta) = (W_{i+1})^T \nabla_{a_{i+1}} L(\theta)$
4. This calculates the gradient w.r.t all neurons of layer *i*. It uses simple matrix-vector multiplication to achieve this.
5. Now, we have seen a special case applied to the last hidden layer. We must figure out how to make this formula applicable for any generic hidden layer.